StrFormat_

Carefully manage buffer sizes.

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-17

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 8235 bytes

Attack Category	• Malicious Input • Denial of Service
Vulnerability Category	• Buffer Overflow • Multibyte Character
Software Context	• String Formatting
Location	• shlwapi.h
Description	When formatting a number representing a size or a time interval, care should be taken to specify the size of the output buffer correctly. StrFormatByteSize, StrFormatByteSize64, StrFormatKBSize and their implementations convert a number representing a size in bytes into a string describing the approximate number of bytes expressed in appropriate units. StrFromTimeInterval converts a time interval, specified in milliseconds, to a string expressing the approximate time interval in appropriate units. Like all functions that create and return a string, if the buffer size parameter is specified to be larger than the actual size of the result buffer, then the function can write beyond the end of the buffer, introducing a reliability problem and security vulnerability. It is particularly easy to specify the buffer size incorrectly when working with Unicode functions, since the buffer size must be specified as a number of characters rather than a number of bytes. Specifying the buffer size as a number of bytes will mistakenly indicate a buffer twice as large as what is actually available.
APIs	**Function Name** / **Comments** StrFormatByteSize / StrFormatByteSize, StrFormatByteSize64 and their implementations convert a numeric value into a

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

		string that represents the number expressed as a size value in bytes, kilobytes, megabytes, or gigabytes, depending on the size.
	StrFormatByteSizeA	ASCII implementation of StrFormatByteSize
	StrFormatByteSizeW	Unicode implementation of StrFormatByteSize and StrFormatByteSize64
	StrFormatByteSize64	StrFormatByteSize, StrFormatByteSize64 and their implementations convert a numeric value into a string that represents the number expressed as a size value in bytes, kilobytes, megabytes, or gigabytes, depending on the size.
	StrFormatByteSize64A	ASCII implementation of StrFormatByteSize64
	StrFormatKBSize	StrFormatKBSize and its implementations convert a numeric value into a string that represents the number expressed as a size value in kilobytes.
	StrFormatKBSizeA	ASCII implementation of StrFormatKBSize
	StrFormatKBSizeW	Unicode implementation of StrFormatKBSize
	StrFromTimeInterval	StrFromTimeInterval converts a time interval, specified in milliseconds, to a string.
	StrFromTimeIntervalA	ACII implementation of StrFromTimeInterval
	StrFromTimeIntervalW	Unicode implementation of StrFromTimeInterval
Method of Attack	If attacker is able to control the input size or interval to be formatted, then they may be able to arrange for the formatting function to overwrite the end of its output buffer. The data written beyond the end of the buffer is not arbitrary, and there is a limit to	

	how far beyond the end of the buffer the functions can be coerced to write. As a result, the risk of the attacker seizing control of the program and executing arbitrary code is less than it would be for some other types of buffer overflow attacks. However, in some cases siezing control could in still be possible. Even if gaining complete control is not immediately feasiblet, this type of bug may allow an attacker to cause the program to fail or behave in a unexpected fashion, which might cause a denial of service or lead to other vulnerabilities being exposed.
Exception Criteria	
Solutions	

Solution Applicability	Solution Description	Solution Efficacy
Whenever the indentified functions are called.	Always specify the output buffer size as a number of characters. If the buffer is allocated on the stack, this may be achieved by specifying the buffer size as: sizeof(buffer)/ sizeof(buffer[0]). If the buffer is allocated on the heap, then the constant used to specify the number of characters to allocate should also be used to specify the size of the buffer to the formatting function.	Effective.

Signature Details	LPTSTR StrFormatByteSizeA(DWORD dw, LPSTR pszBuf, UINT cchBuf); LPTSTR StrFormatByteSizeW(LONGLONG qdw, LPWSTR pwszBuf, UINT cchBuf);

	LPTSTR StrFormatByteSize64(LONGLONG qdw, LPTSTR pszBuf, UINT uiBufSize); LPTSTR StrFormatKBSize(LONGLONG qdw, LPTSTR pszBuf, UINT uiBufSize); int StrFromTimeInterval(LPTSTR pszOut, UINT cchMax, DWORD dwTimeMS, int digits);

Examples of Incorrect Code	`TCHAR pszOut[6]; // Note: this is actually too small given the number of digits specified below`
	`DWORD dwTimeMS = 12345;`
	`int digits 5;`
	
	`// If we are dealing with wide characters (i.e., Unicode) then the following will be incorrect`
	`// and buffer could overflow given a large value of dwTimeMS`
	`StrFromTimeInterval(pszOut, sizeof(pszOut), dwTimeMS, digits);`

Examples of Corrected Code	`TCHAR pszOut[30];`
	`DWORD dwTimeMS = 12345;`
	`int digits 5;`
	
	`// Buffer size specification is correct even for wide characters`
	`StrFromTimeInterval(pszOut, sizeof(pszOut)/sizeof(pszOut[0]), dwTimeMS, digits);`

Source Reference	• Rough Auditing Tool for Security (RATS)[2]
Recommended Resource	• MSDN reference for Shell String Handling Functions[3]
Discriminant Set	**Operating System**: • Windows (All) **Language**: • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com